

LKFmixer: Exploring Large Kernel Feature For Efficient Image Super-Resolution

Yinggan Tang, Quanwei Hu

arXiv:2508.11391v1 [eess.IV] 15 Aug 2025

Abstract—The success of self-attention (SA) in Transformer demonstrates the importance of non-local information to image super-resolution (SR), but the huge computing power required makes it difficult to implement lightweight models. To solve this problem, we propose a pure convolutional neural network (CNN) model, LKFmixer, which utilizes large convolutional kernel to simulate the ability of self-attention to capture non-local features. Specifically, we increase the kernel size to 31 to obtain the larger receptive field as possible, and reduce the parameters and computations by coordinate decomposition. Meanwhile, a spatial feature modulation block (SFMB) is designed to enhance the focus of feature information on both spatial and channel dimension. In addition, by introducing feature selection block (FSB), the model can adaptively adjust the weights between local features and non-local features. Extensive experiments show that the proposed LKFmixer family outperform other state-of-the-art (SOTA) methods in terms of SR performance and reconstruction quality. In particular, compared with SwinIR-light on Manga109 dataset, LKFmixer-L achieves 0.6dB PSNR improvement at $\times 4$ scale, while the inference speed is $\times 5$ times faster. The code is available at <https://github.com/Supereeee/LKFmixer>.

Index Terms—Efficient super-resolution, convolutional neural network, large convolutional kernel, large receptive field.

I. INTRODUCTION

THE process of recovering its high-resolution (HR) counterpart from a low-resolution (LR) image is called image super-resolution (SR). Recently, the excellent performance of SR models based on convolutional neural networks (CNNs) benefits from the increased network complexity [1], [2]. However, these models face challenges when deployed in resource-constrained scenarios, which stimulates people to develop lightweight SR models with fewer parameters, higher efficiency, and faster inference speed [3]. With reduced model complexity, the performance of lightweight SR models highly depends on efficient feature extraction and large receptive field [4]. Nevertheless, the localization of convolution operations in CNN-based SR models lead to limited receptive field though it has powerful local feature extraction ability. In contrast, the self-attention (SA) mechanism in Transformer-based SR models [5] possesses powerful ability to capture non-local features and long-range dependencies, but this comes at the cost of high computational resource demands and slow inference speed. Is there a way that makes CNNs equip with Transformer's ability of capturing non-local features with less computation consumption and inference time? One potential solution may lie in the adoption of larger convolution kernels. Previous studies [6], [7] have demonstrated that increasing the size of

Yinggan Tang and Quanwei Hu are with the School of Electrical Engineering, Yanshan University, Qinhuangdao, Hebei 066004, China. (Email: ygtang@ysu.edu.cn, quanwei1277@163.com)

Corresponding author: Yinggan Tang.

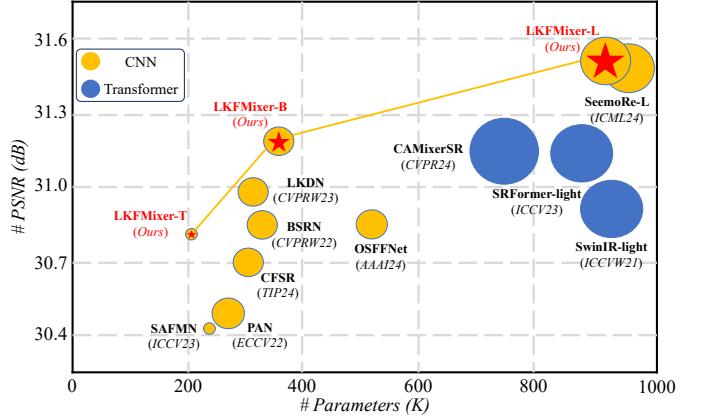


Fig. 1. SR performance and model complexity comparison between our proposed LKFmixer family and other SOTA lightweight models on Manga109 for $\times 4$ SR task. The circle sizes indicate the number of FLOPs.

the convolution kernel can effectively expand the receptive field, thereby enhancing the extraction of non-local features. However, this also leads to a significant increase in model complexity as the model complexity is proportional to the square of the convolution kernel size. To mitigate this issue and reduce the model complexity of CNNs with large convolution kernels, several approaches have been adopted, such as dilated convolution [8] and large kernel decomposition [9]. However, compared with the true large kernel convolutions, these methods often fail to effectively utilize all pixels in the same receptive field and result in feature information loss to a certain extent, which will negatively impact the SR performance and the quality of image reconstruction.

To address the above issues, we propose a novel method called LKFmixer, which applies large convolution kernel in CNN-based lightweight SR models for the purpose of enlarging receptive field while reducing model complexity. To be specific, the size of depth-wise convolution (DWConv) kernel is increased to 31, which is decomposed into DWConv1 \times 31 and DWConv31 \times 1 in series by coordinate decomposition. To further reduce information redundancy across channels and model complexity, we apply partial convolution (PConv) [10], utilizing the decomposed large convolution kernel to extract features from the first quarter of the channels. Based on above design, we propose a partial large kernel block (PLKB), which allows for an increased kernel size with little impact on model complexity and inference time, making it well-suited for lightweight SR models. Additionally, we introduce a feature fusion block (FFB) that aggregates the outputs of DWConv3 \times 3 and PLKB to extract both local and non-local features. By

utilizing FFB, we construct a feature distillation block (FDB) as the core unit for feature extraction and fusion. Meanwhile, we introduce a feature selection block (FSB) to adaptively balance the contribution of local features and non-local features. In addition, in order to pay more attention to spatial and channel information, a spatial feature modulation block (SFMB) is also proposed to improve the SR performance.

As shown in Fig. 1, the proposed LKFMixer family strikes a better trade-off between model complexity and SR performance compared with the recent state-of-the-art (SOTA) lightweight SR model. In summary, our contributions are as follows

- We propose PLKB, which effectively captures non-local features and alleviates the increase in model complexity brought by large convolution kernel.
- By introducing SFMB and FSB, our model adaptively adjusts the contribution of local and non-local features, while simultaneously enhancing spatial and channel information.
- Extensive experiments demonstrate that our proposed LKFMixer family outperforms current SOTA lightweight SR models, achieving superior SR performance while maintaining faster inference speed.

II. RELATED WORK

A. CNN-based Efficient SR

Thanks to the powerful feature extraction capability of CNNs, numerous lightweight SR models based on CNNs have rapidly emerged since the introduction of SRCNN [11]. To reduce model parameters, various variants of standard convolution, such as depth-wise convolution (DWConv) [12] and blueprint separable convolution (BSConv) [13], have been introduced into lightweight SR models. Meanwhile, feature distillation strategy has been employed in IMDN [14], RFDN [15], and BSRN [16] to refine features. Additionally, re-parameterization technique has also been adopted in RepRFN [17] to accelerate inference. In terms of attention mechanism, enhanced spatial attention (ESA) and contrast-aware channel attention (CCA) have been proposed in RFANet [18] and IMDN [14] to improve SR performance with fewer model parameters, respectively. In this paper, we explore the potential of applying large convolution kernel to lightweight SR model.

B. Transformer-based Efficient SR

Transformer [5] exhibits excellent property in modeling long-distance dependencies but suffers from the substantial computational burden imposed by the self-attention (SA) mechanism. To reduce the model complexity, SwinIR [19] introduces sliding local windows to calculate attention weights. ELAN [20] employs group self-attention to improve efficiency. SRFormer [21] leverages permuted self-attention to enjoy the benefit of large window self-attention. CAMixerSR [22] combines convolution with deformable window self-attention to achieve superior SR performance. While these SR models effectively reduce the complexity of self-attention, a significant gap remains in actual inference speed when compared with CNN-based SR models.

C. Large kernel design

Currently, researchers prefer to stack multiple 3×3 convolutions to reduce model parameters. However, it has shown that larger convolution kernels can capture more feature information and significantly expand the receptive field. For example, ConvNeXt [6] employs 7×7 convolution and achieves the same even better performance compared with Swin-Transformer. RepLKNet [7] suggests guidelines for large kernel design and increases the convolution kernel size to 31×31 . PeLK [23] introduces a human-like peripheral convolution, further increasing the kernel size to an impressive 101×101 . These models all highlight the effectiveness of large convolution kernels. However, the substantial computations required by large convolution kernel remain a significant challenge, which will be further explored in this paper.

III. PROPOSED METHOD

In this section, we provide a detailed overview of the various components of the proposed model. Fig. 2 illustrates the overall architecture of LKFMixer, which is primarily composed of three key components. The first component is a 3×3 convolution layer for shallow feature extraction. The second component consists of multiple stacked feature modulation blocks (FMBs) for deep feature extraction. Each FMB is composed of a sequentially arrangement of a feature distillation block (FDB), a spatial feature modulation block (SFMB), and a feature selection block (FSB). The third component is the up-sampler module, which includes a 3×3 convolutional layer followed by a sub-pixel layer [24].

A. Feature Distillation Block

1) *Partial Large Kernel Block:* FasterNet [10] proposed partial convolution (PConv) that performs convolution operations only on partial channels. Inspired by PConv, we extract non-local features by adopting DWConv 31×31 from the input I_{in} on partial channels while the remaining channels keep unchanged. Furthermore, we decompose DWConv 31×31 into two stripe convolutions in series, DWConv 1×31 and DWConv 31×1 , via coordinate decomposition to further reduce model complexity. This process can be expressed as

$$I_{out}^{\alpha C} = \text{DWConv}_{31 \times 1}(\text{DWConv}_{1 \times 31}(I_{in}^{\alpha C})) \quad (1)$$

where α denotes the split factor of the channels, C denotes the number of channels of I_{in} , $I_{in}^{\alpha C}$ represents the first αC channels of I_{in} , and $I_{out}^{\alpha C}$ indicates the output of decomposed large kernel on αC channels. Then, 1×1 convolution is utilized for information fusion after concatenation along the channel dimension. The final output I_{out} of PLKB can be calculated as

$$I_{out} = \text{Conv}_{1 \times 1}(I_{out}^{\alpha C} \otimes I_{in}^{(1-\alpha)C}) \quad (2)$$

where \otimes indicates the channel concatenation operation, and $I_{in}^{(1-\alpha)C}$ denotes the features of remaining unchanged channels.

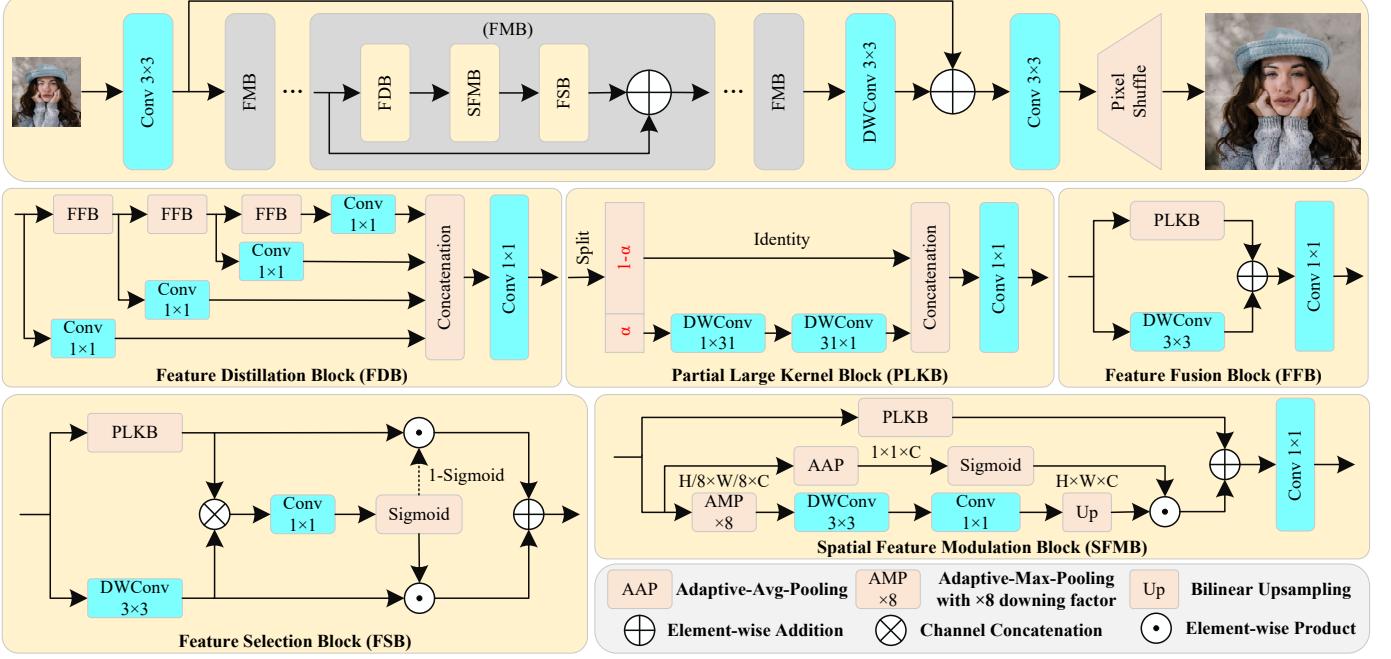


Fig. 2. The overall architecture of LKFMixer, and the detail structure of FDB, SFMB and FSB.

2) *Feature Fusion Block*: We extract local and non-local features via DWConv 3×3 and PLKB, respectively. A 1×1 convolution layer is adopted for information fusion. For the input I_{in} and output I_{out} of FFB, the calculation process can be described as

$$I_{out} = \text{Conv}_{1 \times 1}(\text{DWConv}_{3 \times 3}(I_{in}) + F_{PLKB}(I_{in})) \quad (3)$$

By embedding multiple FFBs into various levels of the distillation structure, rich local and non-local features are gradually refined in the trunk branch. To eliminate redundant information, 1×1 convolutions are used in the distillation branch. The whole process of FDB can be expressed as

$$\begin{aligned} I_r^1, I_d^1 &= FFB_1(I_{in}), \text{Conv}_{1 \times 1}^1(I_{in}) \\ I_r^2, I_d^2 &= FFB_2(I_r^1), \text{Conv}_{1 \times 1}^2(I_r^1) \\ I_r^3, I_d^3 &= FFB_3(I_r^2), \text{Conv}_{1 \times 1}^3(I_r^2) \end{aligned} \quad (4)$$

where I_{in} is the input, I_r^i and I_d^i are the i th layer output of refine branch and distillation branch. FFB_i , $\text{Conv}_{1 \times 1}^i$ ($i = 1, 2, 3$) denote the i th FFB and 1×1 convolution, respectively. Finally, a 1×1 convolution layer is utilized to merge the feature information after channel concatenation, expressed as

$$I_{out} = \text{Conv}_{1 \times 1}(I_d^1 \otimes I_d^2 \otimes I_d^3 \otimes I_r^3) \quad (5)$$

where I_{out} is the output of FDB, and \otimes indicates the channel concatenation operation.

B. Spatial Feature Modulation Block

Both channel and spatial information are critical for SR task. To this end, we design a spatial feature modulation block (SFMB). It includes a spatial branch that captures low-frequency spatial features through downsampling, along with a

channel attention branch that focuses on channel information. The computational process is given by

$$I_s = F_{up}(\text{Conv}_{1 \times 1}(F_{AMP}^{ \times 8}(I_{in}))) \odot F_{sig}(F_{AAP}(I_{in})) \quad (6)$$

where F_{AAP} and $F_{AMP}^{ \times 8}$ are the operation of Adaptive-Avg-Pooling and Adaptive-Max-Pooling with $\times 8$ downing factor along the spatial dimension, respectively. F_{sig} denotes the sigmoid function, F_{up} indicates Bilinear upsampling operation, and \odot denotes element-wise product. By utilizing a 1×1 convolution to fuse the information from the spatial branch and the PLKB module, both non-local features and low-frequency spatial feature are extracted simultaneously. The process is expressed as

$$I_{out} = \text{Conv}_{1 \times 1}(F_{PLKB}(I_{in}) + I_s) \quad (7)$$

where I_{in} and I_{out} are the input and output of SFMB, F_{PLKB} is the PLKB module.

C. Feature Selection Block

To adaptively adjust the weights of local and non-local features, we develop a feature selection block (FSB). First, the features from the PLKB and the DWConv 3×3 are concatenated along the channel dimension and aggregated by a 1×1 convolution. The fused features go through a sigmoid function to obtain the attention weights β . The features of DWConv 3×3 branch are weighted by β while the features of PLKB branch are weighted by $1 - \beta$. The two branches influence each other, adjusting the weight between non-local and local features. Finally, the information from the two branches is aggregated via summation operation. The entire process is as follows

$$I_{PLKB}, I_{DWConv3} = F_{PLKB}(I_{in}), \text{DWConv}_{3 \times 3}(I_{in})$$

$$\beta = F_{sig}(F_{Conv}_{1 \times 1}(I_{PLKB} \otimes I_{DWConv3})) \quad (8)$$

$$I_{out} = I_{DWConv3} \odot \beta + I_{PLKB} \odot (1 - \beta)$$

where I_{in} and I_{out} are the input and output of SFB, \otimes and \odot denote channel concatenation and element-wise product, respectively.

IV. EXPERIMENTS

A. Datasets and indices

Following previous works [28], [29], [34], DIV2K [36] and Flickr2K [2] datasets are used for training. LR images are generated from HR images through bicubic downsampling. Five commonly used datasets-Set5 [37], Set14 [38], BSD100 [39], Urban100 [40] and Manga109 [41] are adopted for testing. Meanwhile, PSNR and SSIM results are calculated on the Y-channel in the YCbCr color space.

B. Implementation Details

The training HR images are decomposed into 480×480 small pieces using sliding window slicing operation for faster training, with random rotation, horizontal and vertical flipping for data augment. The cropped LR patch size is 48×48 , and the batch size is 64. Similar to [29], we optimize L_1 loss and FFT loss using Adam [42] optimizer for 1000K iterations. The initial and minimum learning rates are set to 1×10^{-3} and 1×10^{-6} , which are updated according to the Cosine Annealing scheme. We conduct all experiments with the PyTorch framework on an NVIDIA RTX 3090 GPU. The number of channels and FMBs are set to $\{40, 48, 64\}$ and $\{6, 8, 12\}$ for {LKFMixer-T, LKFMixer-B, LKFMixer-L}, respectively. For all LKFMixer models, the channel split factor α and large kernel size in PLKB are set to 0.25 and 31, respectively.

C. Comparison with state-of-the-art methods

To evaluate the performance of the LKFMixer family, we compare it with several SOTA lightweight SR models, including PAN [25], SAFMN [26], SMFANet [28], SeeMoRe [29], ShuffleMixer [30], BSRN [16], VapSR [31], LKDN [33], CFSR [33], OSSFNet [34], SwinIR-light [19], ELAN-light [20], SRFormer-light [21], MAN [27], MambaIR-light [35], and CAMixerSR-light [22].

1) *Quantitative comparison:* Table I presents the quantitative comparison results in terms of model parameters, computations, PSNR, and SSIM. As shown, our method almost achieves the best SR performance across all test datasets and scales. For instance, LKFMixer-T mostly outperforms SeeMoRe-T and achieves 0.08dB PSNR improvement on the Urban100 dataset at $\times 2$ scale. LKFMixer-L shows remarkable superiority over several models, including SeeMoRe-L, CAMixerSR-light, and MambaIR-light, achieving 0.04dB, 0.34dB, and 0.26dB PSNR improvement on the Manga109 dataset at $\times 4$ scale, respectively. These results demonstrate the effectiveness of the LKFMixer family across varying model complexities.

2) *Qualitative comparison:* Fig. 5 illustrates the reconstruction results of LKFMixer family compared with other lightweight SR methods. As shown, our method consistently delivers superior reconstruction quality across three different model complexities. Specifically, only LKFMixer-T can recover the original structure of the image (e.g., Barbara from

Set14), LKFMixer-B can recover more complete structural features with less distortion (e.g., Img-060 from Urban100), LKFMixer-L has a stronger reconstruction capability and can recover more texture details (e.g., Img-024 and Img-073 from Urban100). These results highlight the effectiveness of large kernel convolution for image reconstruction.

3) *LAM and ERF comparison:* The local attribution map (LAM) [43] is used to visualize the pixel range involved in image reconstruction, which can also be quantified using the diffusion index (DI). A larger DI indicates that more pixels are utilized for reconstruction. As shown in Fig. 3, the DI value for LKFMixer-L is higher than that of other methods, suggesting that LKFMixer-L captures a larger number of pixels and thus has a broader receptive field. The effective receptive field (ERF) [7], which represents the actual receptive field that the model can access, is compared in Fig. 4. The results indicate that LKFMixer-L has a larger ERF compared to the other models. These observations further validate the effectiveness of our large kernel convolution design in enhancing the receptive field.

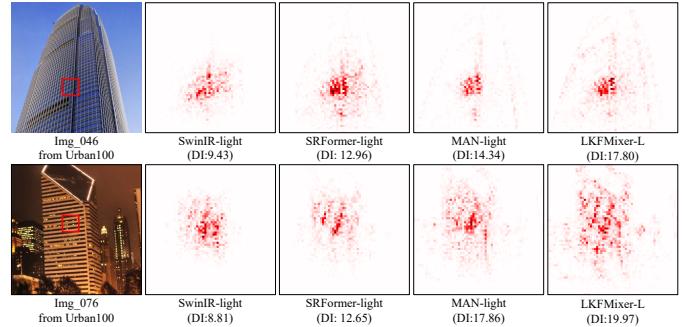


Fig. 3. Comparison of local attribution maps (LAMs) [43] between LKFMixer-L with other lightweight SR models.



Fig. 4. Comparison of effective receptive field (ERF) [7] between LKFMixer-L with other lightweight SR models.

4) *Memory and inference time comparison:* To thoroughly verify the efficiency of the proposed model, we compare GPU memory consumption and inference time with other methods in Table II. In terms of GPU memory consumption, the proposed model approximates other CNN-based models but is much lower than Transformer-based models (only 45% of SwinIR-light). Regarding inference speed, the proposed model is equal to or slightly slower than CNN-based models but is much faster than Transformer-based models (almost $\times 5$ faster than SwinIR-light). The results demonstrate the efficiency of the proposed model.

5) *Visual perception comparison:* To further prove the effectiveness of LKFMixer, we adopt perceptual metric LPIPS [45] to evaluate the visual quality of the reconstructed images,

TABLE I

QUANTITATIVE COMPARISON BETWEEN THE PROPOSED METHOD AND OTHER LIGHTWEIGHT SR METHODS ON BENCHMARK DATASETS. THE BEST AND SECOND BEST RESULTS ARE HIGHLIGHTED IN RED AND BLUE RESPECTIVELY. #FLOPS IS MEASURED BY RECOVERING A 1280×720 HR IMAGE.

Scale	Method	Years	#Params [K]	#Flops [G]	Set5 PSNR / SSIM	Set14 PSNR / SSIM	BSD100 PSNR / SSIM	Urban100 PSNR / SSIM	Manga109 PSNR / SSIM
$\times 2$	PAN [25]	ECCV2020	261	70.6	38.00 / 0.9605	33.59 / 0.9181	32.18 / 0.8997	32.01 / 0.9273	38.70 / 0.9773
	SAFMN [26]	ICCV2023	228	52	38.00 / 0.9605	33.54 / 0.9177	32.16 / 0.8995	31.84 / 0.9256	38.71 / 0.9771
	MAN-tiny [27]	CVPRW2024	134	30	37.93 / 0.9604	33.48 / 0.9171	32.13 / 0.8993	31.75 / 0.9250	38.58 / 0.9770
	SMFANet [28]	ECCV2024	186	41	38.08 / 0.9607	33.65 / 0.9186	32.22 / 0.9002	32.20 / 0.9282	39.11 / 0.9779
	SeemoRe-T [29]	ICML2024	220	46.1	38.06 / 0.9608	33.65 / 0.9186	32.23 / 0.9004	32.22 / 0.9286	39.01 / 0.9777
	LKFMixe-T	Ours	194	41.1	38.09 / 0.9609	33.73 / 0.9190	32.24 / 0.9007	32.30 / 0.9300	39.11 / 0.9781
	ShuffleMixer [30]	NIPS2022	394	91	38.01 / 0.9606	33.63 / 0.9180	32.17 / 0.8995	31.89 / 0.9257	38.83 / 0.9774
	BSRN [16]	CVPRW2022	332	72.2	38.10 / 0.9610	33.74 / 0.9193	32.24 / 0.9006	32.34 / 0.9303	39.14 / 0.9782
	VapSR [31]	ECCV2022	343	78.2	38.08 / 0.9612	33.77 / 0.9195	32.27 / 0.9011	32.45 / 0.9316	39.09 / 0.9782
	LKDNN [32]	CVPRW2023	304	69.2	38.12 / 0.9611	33.90 / 0.9202	32.27 / 0.9010	32.53 / 0.9322	39.19 / 0.9784
	CFSR [33]	TIP2024	291	62.6	38.07 / 0.9607	33.74 / 0.9192	32.24 / 0.9005	32.28 / 0.9300	39.00 / 0.9778
	OSFFNet [34]	AAAI2024	516	83.2	38.11 / 0.9610	33.72 / 0.9190	32.29 / 0.9012	32.67 / 0.9331	39.09 / 0.9780
	LKFMixe-B	Ours	357	75.9	38.21 / 0.9613	33.94 / 0.9209	32.33 / 0.9017	32.75 / 0.9337	39.29 / 0.9786
	SwinIR-light [19]	ICCVW2021	910	244	38.14 / 0.9611	33.86 / 0.9206	32.31 / 0.9012	32.76 / 0.9340	39.12 / 0.9783
	ELAN-light [20]	ECCV2022	621	203	38.17 / 0.9611	33.94 / 0.9207	32.30 / 0.9012	32.76 / 0.9340	39.12 / 0.9783
	SRFormer-light [21]	ICCV2023	853	236	38.23 / 0.9613	33.94 / 0.9209	32.36 / 0.9019	32.91 / 0.9353	39.28 / 0.9785
	MAN-light [27]	CVPRW2024	823	184	38.20 / 0.9613	33.95 / 0.9214	32.36 / 0.9022	32.92 / 0.9364	39.40 / 0.9786
	MambaIR-light [35]	ECCV2024	859	198.1	38.16 / 0.9610	34.00 / 0.9212	32.34 / 0.9017	32.92 / 0.9356	39.31 / 0.9779
	SeemoRe-L [29]	ICML2024	953	202	38.27 / 0.9616	34.01 / 0.9210	34.35 / 0.9018	32.87 / 0.9344	39.49 / 0.9790
	LKFMixe-L	Ours	906	193	38.31 / 0.9617	34.08 / 0.9223	32.42 / 0.9029	33.13 / 0.9371	39.57 / 0.9791
$\times 3$	PAN [25]	ECCV2020	261	39.1	34.10 / 0.9271	30.36 / 0.8423	29.11 / 0.8050	28.11 / 0.8511	33.61 / 0.9448
	SAFMN [26]	ICCV2023	233	23	34.34 / 0.9267	30.33 / 0.8418	29.08 / 0.8048	27.95 / 0.8474	33.52 / 0.9437
	MAN-tiny [27]	CVPRW2024	141	14	34.24 / 0.9259	30.25 / 0.8405	29.04 / 0.8046	27.85 / 0.8465	33.28 / 0.9426
	SMFANet [28]	ECCV2024	191	19	34.42 / 0.9274	30.41 / 0.8430	29.16 / 0.8063	28.22 / 0.8523	33.96 / 0.9460
	SeemoRe-T [29]	ICML2024	225	21	34.46 / 0.9276	30.44 / 0.8445	29.15 / 0.8063	28.27 / 0.8538	33.92 / 0.9460
	LKFMixe-T	Ours	199	18.8	34.51 / 0.9279	30.46 / 0.8447	29.18 / 0.8077	28.27 / 0.8541	33.97 / 0.9464
	ShuffleMixer [30]	NIPS2022	415	43	34.40 / 0.9272	30.37 / 0.8423	29.12 / 0.8051	28.08 / 0.8498	33.69 / 0.9448
	BSRN [16]	CVPRW2022	340	33	34.46 / 0.9277	30.47 / 0.8449	29.18 / 0.8068	28.39 / 0.8567	34.05 / 0.9471
	VapSR [31]	ECCV2022	351	35.6	34.52 / 0.9284	30.53 / 0.8452	29.19 / 0.8077	28.43 / 0.8583	33.96 / 0.9469
	LKDNN [32]	CVPRW2023	311	31.6	34.54 / 0.9285	30.52 / 0.8455	29.21 / 0.8078	28.50 / 0.8601	34.08 / 0.9475
	CFSR [33]	TIP2024	298	28.5	34.50 / 0.9279	30.44 / 0.8437	29.16 / 0.8066	28.29 / 0.8553	33.86 / 0.9462
	OSFFNet [34]	AAAI2024	524	37.8	34.58 / 0.9287	30.48 / 0.8450	29.21 / 0.8080	28.49 / 0.8595	34.00 / 0.9472
	LKFMixe-B	Ours	364	34.4	34.59 / 0.9287	30.57 / 0.8463	29.25 / 0.8095	28.58 / 0.8604	34.26 / 0.9481
	SwinIR-light [19]	ICCVW2021	918	114	34.62 / 0.9289	30.54 / 0.8463	29.20 / 0.8082	28.66 / 0.8624	33.98 / 0.9478
	ELAN-light [20]	ECCV2022	629	90.1	34.61 / 0.9288	30.55 / 0.8463	29.21 / 0.8081	28.69 / 0.8624	34.00 / 0.9478
	SRFormer-light [21]	ICCV2023	861	105	34.67 / 0.9296	30.57 / 0.8469	29.26 / 0.8099	28.81 / 0.8655	34.19 / 0.9489
	MAN-light [27]	CVPRW2024	832	82.7	34.66 / 0.9293	30.60 / 0.8478	29.29 / 0.8105	28.87 / 0.8671	34.36 / 0.9492
	MambaIR-light [35]	ECCV2024	867	88.7	34.72 / 0.9296	30.63 / 0.8475	29.29 / 0.8099	29.00 / 0.8689	34.39 / 0.9495
	SeemoRe-L [29]	ICML2024	959	90.5	34.72 / 0.9297	30.60 / 0.8469	29.29 / 0.8101	28.86 / 0.8653	34.53 / 0.9496
	LKFMixe-L	Ours	915	86.8	34.76 / 0.9301	30.66 / 0.8485	29.34 / 0.8118	28.97 / 0.8677	34.63 / 0.9503
$\times 4$	PAN [25]	ECCV2020	272	28.2	32.13 / 0.8948	28.61 / 0.7822	27.59 / 0.7363	26.11 / 0.7854	30.51 / 0.9095
	SAFMN [26]	ICCV2023	240	14	32.18 / 0.8948	28.60 / 0.7813	27.58 / 0.7359	25.97 / 0.7809	30.43 / 0.9063
	MAN-Tiny [27]	CVPRW2024	150	8.4	32.07 / 0.8930	28.53 / 0.7801	27.51 / 0.7345	25.84 / 0.7786	30.18 / 0.9047
	SMFANet [28]	ECCV2024	197	11	32.25 / 0.8956	28.71 / 0.7833	27.64 / 0.7377	26.18 / 0.7862	30.82 / 0.9104
	SeemoRe-T [29]	ICML2024	232	11	32.31 / 0.8965	28.72 / 0.7840	27.65 / 0.7384	26.23 / 0.7883	30.82 / 0.9107
	LKFMixe-T	Ours	207	11	32.34 / 0.8965	28.74 / 0.7843	27.67 / 0.7391	26.23 / 0.7890	30.76 / 0.9102
	ShuffleMixer [30]	NIPS2022	411	28	32.21 / 0.8953	28.66 / 0.7827	27.61 / 0.7366	26.08 / 0.7835	30.65 / 0.9093
	BSRN [16]	CVPRW2022	352	19.2	32.35 / 0.8966	28.73 / 0.7847	27.65 / 0.7387	26.27 / 0.7908	30.84 / 0.9123
	VapSR [31]	ECCV2022	342	19.8	32.38 / 0.8978	28.77 / 0.7852	27.68 / 0.7398	26.35 / 0.7941	30.86 / 0.9129
	LKDNN [32]	CVPRW2023	322	18.4	32.39 / 0.8979	28.79 / 0.7859	27.69 / 0.7402	26.42 / 0.7965	30.97 / 0.9140
	CFSR [33]	TIP2024	307	17.5	32.33 / 0.8964	28.73 / 0.7842	27.63 / 0.7381	26.21 / 0.7897	30.72 / 0.9111
	OSFFNet [34]	AAAI2024	537	22	32.39 / 0.8976	28.75 / 0.7852	27.66 / 0.7393	26.36 / 0.7950	30.84 / 0.9125
	LKFMixe-B	Ours	373	19.9	32.46 / 0.8982	28.85 / 0.7865	27.75 / 0.7415	26.48 / 0.7962	31.17 / 0.9148
	SwinIR-light [19]	ICCVW2021	930	65	32.44 / 0.8976	28.77 / 0.7858	27.69 / 0.7406	26.47 / 0.7980	30.92 / 0.9151
	ELAN-light [20]	ECCV2022	640	54.1	32.43 / 0.8975	28.78 / 0.7858	27.69 / 0.7406	26.54 / 0.7982	30.92 / 0.9150
	SRFormer-light [21]	ICCV2023	873	63	32.51 / 0.8988	28.82 / 0.7872	27.73 / 0.7422	26.67 / 0.8032	31.17 / 0.9165
	MAN-light [27]	CVPRW2024	840	47.1	32.50 / 0.8988	28.87 / 0.7885	27.77 / 0.7429	26.70 / 0.8052	31.25 / 0.9170
	CAMixerSR [22]	CVPR2024	765	77.9	32.51 / 0.8988	28.82 / 0.7870	27.72 / 0.7416	26.63 / 0.8012	31.18 / 0.9166
	MambaIR-Light [35]	ECCV2024	879	50.6	32.51 / 0.8993	28.85 / 0.7876	27.75 / 0.7423	26.75 / 0.8051	31.26 / 0.9175
	SeemoRe-L [29]	ICML2024	969	51.4	32.51 / 0.8990	28.92 / 0.7888	27.78 / 0.7428	26.79 / 0.8046	31.48 / 0.9181
	LKFMixe-L	Ours	927	49.5	32.71 / 0.9008	28.95 / 0.7892	27.83 / 0.7443	26.85 / 0.8069	31.52 / 0.9191

and the results are shown in Table IV. The lower the LPIPS is, the more realistic the recovered image is. Compared with other lightweight Transformer-based methods, our method almost achieves the lowest LPIPS values on all three datasets, except for slightly higher than SRFormer-light [21] on the Urban100 dataset. This proves that the images recovered by the proposed

LKFMixe exhibit superior visual perception and quality.

Meanwhile, to verify the generalization of the proposed model, we compare LKFMixe-L with other SR methods in real-world images, and the results are shown in Fig. 6. ESRGAN [44] excels in recovering more realistic images by using perceptual loss. However, our LKFMixe-L and other

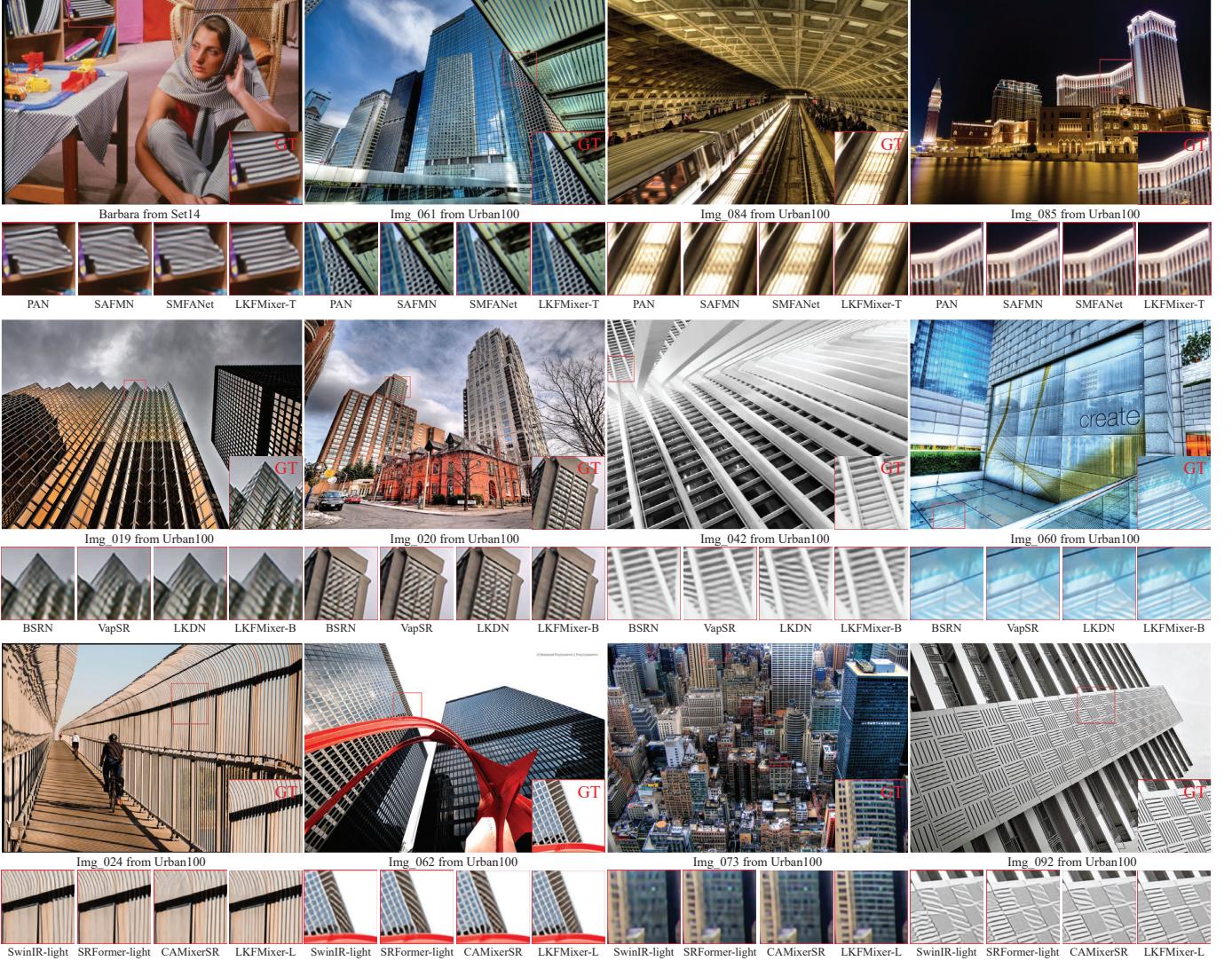


Fig. 5. Visual comparisons between LKFMixer family with other SOTA lightweight models for $\times 4$ SR task. Zoom in for best view.

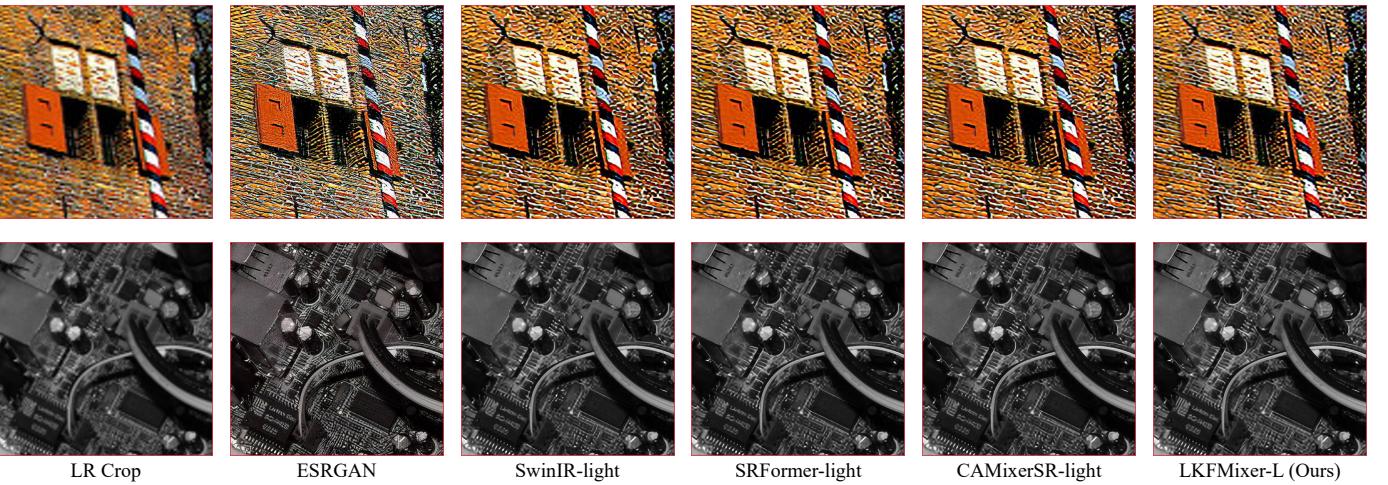


Fig. 6. Visual comparison with ESRGAN [44], SwinIR-light [19], SRFormer-light [21], CAMixerSR-light [22], and the proposed LKFMixer-L on **real-word images** at $\times 4$ upscale.

Transformer-based SR models, tends to produce smoother images while still effectively recovering the overall structure

TABLE II
COMPARISON OF AVERAGE INFERENCE TIME AND GPU MEMORY CONSUMPTION ACROSS 100 SAMPLES ON 3090 GPU.

Input	Scale	Method	#GPU Mem.(M)	#Avg. Time(ms)
[512, 512] × 4		PAN	1196.09	51.72
		SeemoRe-T	365.02	69.27
		LKFmixer-T (Ours)	405.42	61.02
		VapSR	404.36	109.41
		LKDNN	1176.29	68.93
		LKFmixer-B (Ours)	486.05	98.51
		SwinIR-light	1451.48	1000.89
		SRFormer-light	1303.83	998.50
		CAMixerSR	1489.17	311.72
		SeemoRe-L	470.94	232.82
		LKFmixer-L (Ours)	648.21	194.43

and texture details. These results highlight that LKFmixer-L has a robust ability to recover image details, even when processing real-world images with potentially unknown or complex degradation patterns.

V. ANALYSIS AND DISCUSSION

In this section, we conduct extensive ablation experiments to evaluate the effectiveness of the internal modules and large kernel design, as shown in Table III. For the three modules FDB, SFMB and FSB, the model performance decreases when one of them is individually removed, indicating the effectiveness of each module. Next, we provide a detailed analysis of their internal design.

A. Effectiveness of the PLKB

Considering that the large convolution kernel is the core of PLKB, we first replace the serial strip convolutions with DWConv31×31, and the model performance remains unchanged or even decreases. However, the model parameters increase by 115.5%, and the inference time increases by 97.7%, which proves that strip convolution is more suitable for lightweight models. To investigate the impact of kernel size on model performance, we gradually increase the kernel size from 3 to 41, and the results are shown in Table V. The overall model performance gradually improves and peaks when the kernel size reaches 31, but the model performance declines when the kernel size is further increased to 41. Importantly, the increase in kernel size does not lead to a significant rise in model parameters and inference time, thanks to the coordinate decomposition and partial channel design.

We also analyze the relationship between kernel size and receptive field utilizing LAM and ERF, as shown in Fig. 7 and Fig. 8, respectively. It is evident that the receptive field increases as the kernel size grows and reaches maximum when the kernel size is 41. Taking the model performance into consideration, we ultimately select the kernel size of 31 for the PLKB. Additionally, we replace PLKB with the WSA (window-based self-attention) mechanism [19], which leads to a significant improvement in model performance. However, this comes at the cost of 81.8% increase in model parameters, 206.6% increase in GPU memory consumption, and notably ×13 times increase in inference time.

We further analyze how the size of the convolution kernel in PLKB impacts the extracted feature maps, and the results

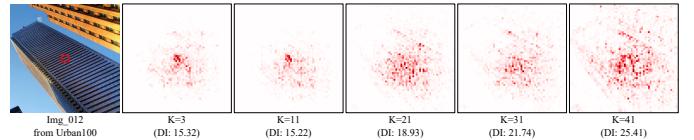


Fig. 7. LAM comparison of different kernel size in PLKB.

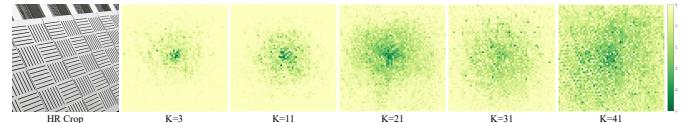


Fig. 8. ERF analysis of different kernel size in PLKB.

are shown in Fig. 9. When the convolution kernel size is 3, the output feature map primarily contains high-frequency details such as edges and textures. This highlights the small convolution kernel's strong ability to capture local features. As the kernel size increases to 31, the feature map begins to include more low-frequency information, such as image contours, while still retaining high-frequency details. This demonstrates the large convolution kernel's ability to capture non-local features. However, when the kernel size is increased further to 41, the high-frequency details become more pronounced, while the low-frequency contour information diminishes. The change in the feature map is consistent with the observed variation in SR performance as the kernel size increases, further validating the effectiveness of larger convolution kernels in enhancing image reconstruction capabilities.

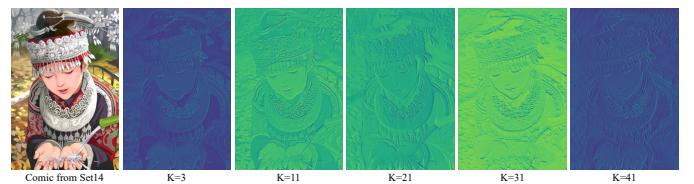


Fig. 9. Feature map visualization of PLKB output with different kernel sizes.

To distinguish the receptive field differences between DWConv31×31 and the decomposed DWConv1×31+31×1, the LAM [43] results of them are shown in Fig. 10. The LAM range for both convolutions is nearly identical, suggesting that the structure of the convolution kernel has minimal impact on the receptive field as long as the kernel size remains unchanged.

In addition, we investigate the differences in feature maps extracted by the two convolution kernel structures, and the results are shown in Fig. 11. The features extracted by DWConv1×31 and DWConv31×1 are almost identical. However, slight differences can be observed in the details (such as leaf and clothing textures) due to the strip structure and operation logic of the convolution kernels. Since DWConv1×31+31×1 consists of two serial strip convolutions, the features it extracts synthesize those of DWConv1×31 and DWConv31×1, with rich texture details while preserving overall contour features. On the other hand, DWConv31×31 can also extract low-frequency contour information and high-frequency edge or texture fea-

TABLE III

ABLATION STUDY BASED ON LKFMIXER-B AT $\times 4$ SCALE. METRICS ARE ALSO MEASURED ON THE Y-CHANNEL. “A→B” IS THE OPERATION OF REPLACING A WITH B, AND “A→NONE” IS THE OPERATION OF REMOVING A. “-” DENOTES REMAIN UNCHANGED. WSA REPRESENTS THE WINDOW-BASED SELF-ATTENTION USED IN SWINIR [19].

Ablation	Variant	#Params [K]	#Flops [G]	#GPU Mem. [M]	#Time [ms]	Set5 PSNR / SSIM	Set14 PSNR / SSIM	BSD100 PSNR / SSIM	Urban100 PSNR / SSIM	Manga109 PSNR / SSIM	DIV2K100 PSNR / SSIM
Baseline	LKFMixer-B	373	19.9	486.05	98.51	32.46 / 0.8982	28.85 / 0.7865	27.75 / 0.7415	26.48 / 0.7962	31.17 / 0.9148	30.71 / 0.8432
FMB	FDB → None	155	7.6	483.63	40.91	32.09 / 0.8936	28.57 / 0.7804	27.57 / 0.7360	25.90 / 0.7783	30.27 / 0.9040	30.42 / 0.8371
	SFMB → None	306	17.3	485.77	81.00	32.43 / 0.8979	28.81 / 0.7860	27.72 / 0.7409	26.45 / 0.7949	30.99 / 0.9126	30.61 / 0.8420
	FSB → None	307	16.1	485.78	74.52	32.40 / 0.8978	28.80 / 0.7859	27.73 / 0.7408	26.39 / 0.7935	31.03 / 0.9127	30.66 / 0.8425
PLKB	$1 \times 31 + 31 \times 1 \rightarrow 31 \times 31$	804	44.7	487.69	194.79	32.45 / 0.8978	28.82 / 0.7862	27.74 / 0.7414	26.48 / 0.7964	31.09 / 0.9143	30.69 / 0.8432
	PLKB → WSA	678	49.3	1490.03	1354.31	32.52 / 0.8987	28.89 / 0.7880	27.77 / 0.7424	26.59 / 0.7996	31.32 / 0.9169	30.74 / 0.8440
FFB	DWConv3 → None	361	19.3	485.99	85.99	32.42 / 0.8975	28.82 / 0.7861	27.74 / 0.7413	26.48 / 0.7965	31.11 / 0.9141	30.69 / 0.8429
	PLKB → None	298	15.6	485.73	67.81	32.36 / 0.8974	28.77 / 0.7854	27.72 / 0.7404	26.36 / 0.7928	30.98 / 0.9129	30.64 / 0.8420
SFMB	PLKB → None	348	18.5	485.94	86.95	30.61 / 0.8609	27.41 / 0.7453	26.88 / 0.7103	24.31 / 0.7085	26.72 / 0.8131	29.08 / 0.9783
	Channel attention → None	373	19.8	486.05	96.65	32.45 / 0.8981	28.83 / 0.7864	27.74 / 0.7415	26.51 / 0.7969	31.06 / 0.9143	30.70 / 0.8433
	Spatial branch → None	350	19.7	485.95	92.14	32.35 / 0.8970	28.79 / 0.7859	27.72 / 0.7406	26.36 / 0.7929	30.96 / 0.9127	30.67 / 0.8425
	Downsampling factor 8 → 4	373	19.9	486.05	99.46	32.42 / 0.8977	28.81 / 0.7862	27.73 / 0.7410	26.45 / 0.7953	31.11 / 0.9140	30.68 / 0.8424
	Downsampling factor 8 → 2	373	20.2	486.05	101.96	32.46 / 0.8979	28.84 / 0.7861	27.74 / 0.7411	26.49 / 0.7964	31.02 / 0.9140	30.69 / 0.8428
FSB	Addition → Concatenation	391	20.9	486.12	99.85	32.43 / 0.8977	28.84 / 0.7868	27.75 / 0.7417	26.48 / 0.7963	31.09 / 0.9144	30.69 / 0.8433
	Concatenation → Addition	354	18.7	485.98	97.19	32.45 / 0.8980	28.81 / 0.7862	27.73 / 0.7410	26.45 / 0.7961	31.08 / 0.9141	30.69 / 0.8428
	1-Sigmoid → Sigmoid	-	-	-	-	32.45 / 0.8981	28.82 / 0.7861	27.73 / 0.7413	26.46 / 0.7957	31.10 / 0.9139	30.69 / 0.8431
Loss function	L1 loss → None	-	-	-	-	32.42 / 0.8980	28.82 / 0.7863	27.73 / 0.7410	26.49 / 0.7962	31.10 / 0.9143	30.70 / 0.8429
	FFT loss → None	-	-	-	-	32.39 / 0.8970	28.78 / 0.7857	27.71 / 0.7406	26.39 / 0.7964	30.85 / 0.9125	30.64 / 0.8427

TABLE IV

LPIPS [45] COMPARISON ON SEVERAL BENCHMARK DATASETS. THE LOWER THE LPIPS, THE BETTER THE VISUAL PERCEPTION OF IMAGES.

Datasets	SwinIR-light [19]	SRFormer-light [21]	CAMixerSR-light [22]	LKFMixer-L (Ours)
Urban100	0.2161	0.2109	0.2150	0.2123
Manga109	0.1021	0.1005	0.1009	0.1001
DIV2K100	0.2634	0.2608	0.2595	0.2595

TABLE V

ABALATION STUDY OF LARGE KERNEL SIZE IN PLKB. THE LARGE KERNEL FORM IS $1 \times K$ AND $K \times 1$.

Kernel	#Params [K]	#Flops [G]	#GPU Mem. [M]	#Time [ms]	BSD100 PSNR / SSIM	Urban100 PSNR / SSIM	Manga109 PSNR / SSIM	DIV2K100 PSNR / SSIM
3	346	18.3	485.97	82.35	27.71 / 0.7403	26.40 / 0.7935	31.05 / 0.9135	30.66 / 0.8421
11	354	18.8	486.01	86.71	27.72 / 0.7405	26.44 / 0.7948	31.08 / 0.9139	30.68 / 0.8426
21	363	19.3	486.01	91.64	27.72 / 0.7407	26.45 / 0.7953	31.11 / 0.9144	30.69 / 0.8430
31	373	19.9	486.05	98.51	27.75 / 0.7415	26.48 / 0.7962	31.17 / 0.9148	30.71 / 0.8432
41	382	20.4	486.09	103.81	27.73 / 0.7413	26.45 / 0.7953	30.92 / 0.9129	30.66 / 0.8426

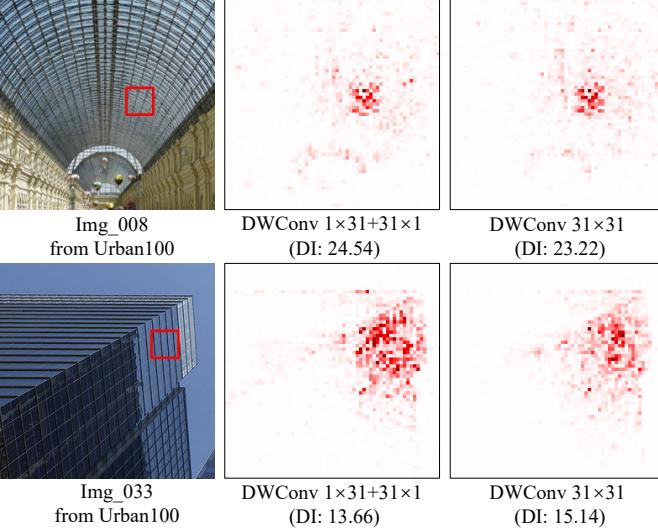


Fig. 10. LAM comparison between large kernel DWConv 31×31 and decomposed series strip large kernel DWConv $1 \times 31 + 31 \times 1$ at $\times 4$ upscale.

tures. Considering the balance between model complexity and inference speed, we conclude that the decomposed strip large

convolution kernel is better suited for the lightweight SR model.

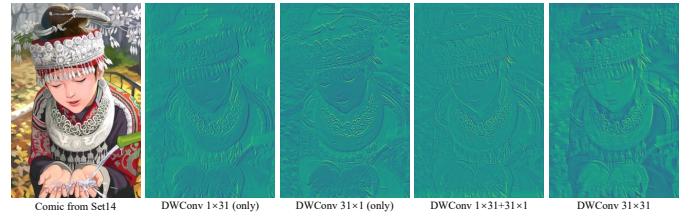


Fig. 11. Feature map visualization of different large kernel output in PLKB.

Meanwhile, we utilize LAM and ERF to analyze the receptive field of PLKB and WSA, as shown in Fig. 12. The results demonstrate that PLKB achieves a larger receptive field, indicating that it can capture non-local features similar to WSA in Transformer-based models, but is more suitable for lightweight SR models.

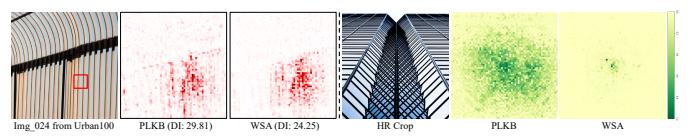


Fig. 12. LAM (left) and ERF (right) comparison of PLKB and WSA [19].

Furthermore, we analyze the influence of channel split factor α on model performance, as shown in Table VI. As α increase, more channels participate in the convolution operation, leading to model parameters and inference time gradually increase, but the model performance does not increase significantly. After comprehensive consideration, we select $\alpha=0.25$ to maintain an optimal balance between model parameters, SR performance, and inference time.

The experimental results in Table VI reveal that as the number of channels increases, the overall model performance improves. To gain a deeper understanding of this relationship, we further analyze the feature map outputs of PLKB under different channel split factors α , and the results are shown in

TABLE VI
ABLATION STUDY OF CHANNEL SPLIT FACTOR α IN PLKB.

α	#Params [K]	#Flops [G]	#GPU Mem. [M]	#Time [ms]	BSD100 PSNR / SSIM	Urban100 PSNR / SSIM	Manga109 PSNR / SSIM	DIV2K100 PSNR / SSIM
0.125	357	19.0	486.01	88.24	27.73 / 0.7410	26.48 / 0.7956	31.10 / 0.9142	30.68 / 0.8427
0.25	373	19.9	486.05	98.51	27.75 / 0.7415	26.48 / 0.7962	31.17 / 0.9148	30.71 / 0.8432
0.5	404	21.6	486.17	116.13	27.74 / 0.7415	26.49 / 0.7965	31.15 / 0.9148	30.71 / 0.8433
0.75	434	23.3	496.76	136.20	27.74 / 0.7420	26.54 / 0.7990	30.98 / 0.9134	30.72 / 0.8438
1	465	25.0	508.88	156.44	27.76 / 0.7419	26.48 / 0.7966	31.12 / 0.9147	30.72 / 0.8438

the Fig. 13. As α increases, more channels are employed for feature extraction, resulting in a feature map that contains more texture details while retaining the essential contour information, which is beneficial for recovering more accurate image details. However, considering both the model complexity and inference time, we determine that $\alpha=0.25$ offers an optimal balance between performance and efficiency, allowing us to maintain the ability to capture detailed features without excessively increasing model complexity.

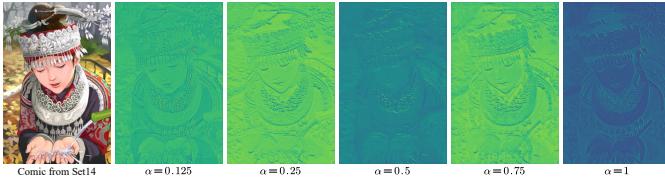


Fig. 13. Feature map visualization of PLKB output with different channel split factor α .

B. Effectiveness of the FFB

PLKB and DWConv3 \times 3 are responsible for capturing non-local and local features, respectively. Fusing the outputs of them is beneficial for obtaining more comprehensive feature information. When either PLKB or DWConv3 \times 3 is individually removed, the model performance decreases, indicating that the features from them are complementary. To further analyze the feature differences they extract, we visualize their feature maps in Fig. 14. It can be observed that DWConv3 \times 3 can extract local details (e.g., whiskers), while PLKB can further capture structural features (e.g., contours). Both local and global features can be effectively preserved after aggregating them.

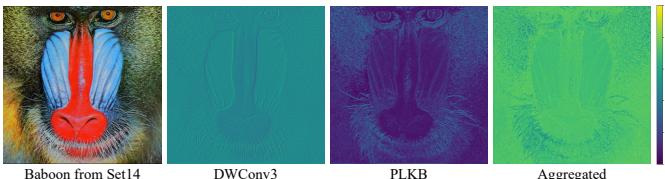


Fig. 14. Visualization of different feature map outputs from FFB.

Following [28], we also utilize power spectral density (PSD) to analyze the frequency-domain features, as shown in Fig. 15. The results indicate that DWConv3 \times 3 tends to capture sparse high-frequency features, while PLKB simultaneously acquires rich high-frequency and low-frequency features in the center of the spectrum. Meanwhile, both low-frequency and high-frequency features can be retained after information aggregation.

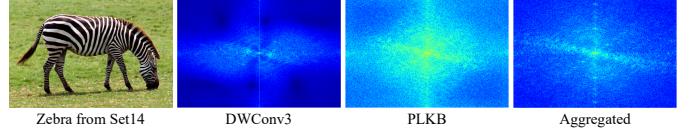


Fig. 15. Visualization of power spectral density (PSD) outputs from FFB, which has shifted the low-frequency component of the spectrum to the center by periodic displacement.

C. Effectiveness of the SFMB

For both PLKB and spatial branches, the model performance decreases significantly when either of them is removed, indicating that non-local features and low-frequency spatial information are essential for SR reconstruction. Similarly, when we remove the channel attention in the spatial branch, the overall model performance decreases, emphasizing the importance of focusing on the salient features in channels. Regarding the downsampling factor in the spatial branch, the model performance drops when we reduce the downsampling factor from 8 to 4 or 2. The reason may be that the abstract information extracted by larger downsampling scale is more helpful to the SR model. Additionally, when we change the fusion mode of the large kernel branch and spatial branch from element-wise addition to channel concatenation, the model complexity increases and the overall model performance decreases. So we utilize element-wise addition for information fusion in SFMB to maintain efficiency and performance.

D. Effectiveness of the SFB

We first analyze the processing strategy of the sigmoid function. If the weight of PLKB branch changes from $1-\beta$ to β while the DWConv3 \times 3 branch is also weighted by β , we observe a degradation in model performance, suggesting that maintaining complementary weights between the branches is beneficial for improving model performance. In addition, we analyze the feature aggregation mode of PLKB and DWConv3 \times 3 features. When the channel concatenation operation is replaced with the element-wise addition, the model performance decreases although the model parameters are slightly reduced, which indicates that the information fusion strategy of channel concatenation is more effective for SFB.

VI. LIMITATIONS AND FUTURE WORK

In this paper, we explore the effectiveness of large kernel convolution in lightweight CNN-based SR models and achieve impressive SR performance, but there are still some limitations. First, the aggregation of non-local and local features plays a crucial role in model performance, but the fusion strategy still needs to be further studied. Second, due to time and computational constraints, this paper does not explore even larger convolution kernel sizes, such as 51 or beyond. Meanwhile, the reason why the SR performance declines when the kernel size exceeds 31 has not been fully analyzed. In future work, we will investigate these issues more thoroughly. Specifically, we will conduct a deeper analysis into why performance declines with larger kernel sizes and explore potential improvements. Moreover, we will continue to explore the application of large

convolution kernels in lightweight SR models to further enhance their performance and efficiency.

VII. CONCLUSION

In this paper, we propose LKFMixer, a pure CNN-based model that leverages large convolution kernels for efficient image SR. By utilizing coordinate decomposition and partial channel convolution, we successfully increase the kernel size to 31, significantly improving SR performance through enhanced non-local feature extraction, while maintaining lower model complexity and faster inference speeds. Extensive experiments demonstrate that the proposed method achieves SOTA SR performance and superior visual quality, with much faster inference speed compared with Transformer-based models. We hope our study will promote the research of large convolution kernels in lightweight SR models.

REFERENCES

- [1] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.
- [2] B. Lim, S. Son, H. Kim, S. Nah, and K. Mu Lee, “Enhanced deep residual networks for single image super-resolution,” in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2017, pp. 136–144.
- [3] B. Ren, Y. Li, N. Mehta, R. Timofte, H. Yu, C. Wan, Y. Hong, B. Han, Z. Wu, Y. Zou *et al.*, “The ninth ntire 2024 efficient super-resolution challenge report,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 6595–6631.
- [4] D. Lee, S. Yun, and Y. Ro, “Partial large kernel cnns for efficient super-resolution,” *arXiv preprint arXiv:2404.11848*, 2024.
- [5] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, “An image is worth 16x16 words: Transformers for image recognition at scale,” *ICLR*, 2021.
- [6] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, “A convnet for the 2020s,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 11 976–11 986.
- [7] X. Ding, X. Zhang, J. Han, and G. Ding, “Scaling up your kernels to 31x31: Revisiting large kernel design in cnns,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 11 963–11 975.
- [8] X. Song, X. Pang, L. Zhang, X. Lu, and X. Hei, “Single image super-resolution with lightweight multi-scale dilated attention network,” *Applied Soft Computing*, p. 112569, 2024.
- [9] M.-H. Guo, C.-Z. Lu, Z.-N. Liu, M.-M. Cheng, and S.-M. Hu, “Visual attention network,” *Computational Visual Media*, vol. 9, no. 4, pp. 733–752, 2023.
- [10] J. Chen, S.-h. Kao, H. He, W. Zhuo, S. Wen, C.-H. Lee, and S.-H. G. Chan, “Run, don’t walk: chasing higher flops for faster neural networks,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 12 021–12 031.
- [11] C. Dong, C. C. Loy, K. He, and X. Tang, “Learning a deep convolutional network for image super-resolution,” in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part IV 13*. Springer, 2014, pp. 184–199.
- [12] F. Chollet, “Xception: Deep learning with depthwise separable convolutions,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258.
- [13] D. Haase and M. Amthor, “Rethinking depthwise separable convolutions: How intra-kernel correlations lead to improved mobilenets,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 14 600–14 609.
- [14] Z. Hui, X. Gao, Y. Yang, and X. Wang, “Lightweight image super-resolution with information multi-distillation network,” in *Proceedings of the 27th ACM international conference on multimedia*, 2019, pp. 2024–2032.
- [15] J. Liu, J. Tang, and G. Wu, “Residual feature distillation network for lightweight image super-resolution,” in *Computer vision–ECCV 2020 workshops: Glasgow, UK, August 23–28, 2020, proceedings, part III 16*. Springer, 2020, pp. 41–55.
- [16] Z. Li, Y. Liu, X. Chen, H. Cai, J. Gu, Y. Qiao, and C. Dong, “Blueprint separable residual network for efficient image super-resolution,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 833–843.
- [17] W. Deng, H. Yuan, L. Deng, and Z. Lu, “Reparameterized residual feature network for lightweight image super-resolution,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 1712–1721.
- [18] J. Liu, W. Zhang, Y. Tang, J. Tang, and G. Wu, “Residual feature aggregation network for image super-resolution,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 2359–2368.
- [19] J. Liang, J. Cao, G. Sun, K. Zhang, L. Van Gool, and R. Timofte, “Swinir: Image restoration using swin transformer,” in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 1833–1844.
- [20] X. Zhang, H. Zeng, S. Guo, and L. Zhang, “Efficient long-range attention network for image super-resolution,” in *European conference on computer vision*. Springer, 2022, pp. 649–667.
- [21] Y. Zhou, Z. Li, C.-L. Guo, S. Bai, M.-M. Cheng, and Q. Hou, “Srformer: Permuted self-attention for single image super-resolution,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 12 780–12 791.
- [22] Y. Wang, Y. Liu, S. Zhao, J. Li, and L. Zhang, “Camixersr: Only details need more ‘attention’,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 25 837–25 846.
- [23] H. Chen, X. Chu, Y. Ren, X. Zhao, and K. Huang, “Pelk: Parameter-efficient large kernel convnets with peripheral convolution,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 5557–5567.
- [24] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, “Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1874–1883.
- [25] H. Zhao, X. Kong, J. He, Y. Qiao, and C. Dong, “Efficient image super-resolution using pixel attention,” in *Computer Vision–ECCV 2020 Workshops: Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*. Springer, 2020, pp. 56–72.
- [26] L. Sun, J. Dong, J. Tang, and J. Pan, “Spatially-adaptive feature modulation for efficient image super-resolution,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 13 190–13 199.
- [27] Y. Wang, Y. Li, G. Wang, and X. Liu, “Multi-scale attention network for single image super-resolution,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 5950–5960.
- [28] M. Zheng, L. Sun, J. Dong, and J. Pan, “Smfanet: A lightweight self-modulation feature aggregation network for efficient image super-resolution,” in *ECCV*, 2024.
- [29] E. Zamfir, Z. Wu, N. Mehta, Y. Zhang, and R. Timofte, “See more details: Efficient image super-resolution by experts mining,” in *Forty-first International Conference on Machine Learning*, 2024.
- [30] L. Sun, J. Pan, and J. Tang, “Shufflemixer: An efficient convnet for image super-resolution,” *Advances in Neural Information Processing Systems*, vol. 35, pp. 17 314–17 326, 2022.
- [31] L. Zhou, H. Cai, J. Gu, Z. Li, Y. Liu, X. Chen, Y. Qiao, and C. Dong, “Efficient image super-resolution using vast-receptive-field attention,” in *European conference on computer vision*. Springer, 2022, pp. 256–272.
- [32] C. Xie, X. Zhang, L. Li, H. Meng, T. Zhang, T. Li, and X. Zhao, “Large kernel distillation network for efficient single image super-resolution,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 1283–1292.
- [33] G. Wu, J. Jiang, J. Jiang, and X. Liu, “Transforming image super-resolution: A conformer-based efficient approach,” *IEEE Transactions on Image Processing*, 2024.
- [34] Y. Wang and T. Zhang, “Ossfnet: Omni-stage feature fusion network for lightweight image super-resolution,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 6, 2024, pp. 5660–5668.
- [35] H. Guo, J. Li, T. Dai, Z. Ouyang, X. Ren, and S.-T. Xia, “Mambair: A simple baseline for image restoration with state-space model,” in *European conference on computer vision*. Springer, 2024, pp. 222–241.
- [36] E. Agustsson and R. Timofte, “Ntire 2017 challenge on single image super-resolution: Dataset and study,” in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2017, pp. 126–135.

- [37] M. Bevilacqua, A. Roumy, C. Guillemot, and M.-L. A. Morel, “Low-complexity single-image super-resolution based on nonnegative neighbor embedding,” in *British Machine Vision Conference (BMVC)*, 2012.
- [38] R. Zeyde, M. Elad, and M. Protter, “On single image scale-up using sparse-representations,” in *Curves and Surfaces: 7th International Conference, Avignon, France, June 24-30, 2010, Revised Selected Papers 7*. Springer, 2012, pp. 711–730.
- [39] D. Martin, C. Fowlkes, D. Tal, and J. Malik, “A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics,” in *Proceedings eighth IEEE international conference on computer vision. ICCV 2001*, vol. 2. IEEE, 2001, pp. 416–423.
- [40] J.-B. Huang, A. Singh, and N. Ahuja, “Single image super-resolution from transformed self-exemplars,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 5197–5206.
- [41] Y. Matsui, K. Ito, Y. Aramaki, A. Fujimoto, T. Ogawa, T. Yamasaki, and K. Aizawa, “Sketch-based manga retrieval using manga109 dataset,” *Multimedia tools and applications*, vol. 76, pp. 21 811–21 838, 2017.
- [42] D. P. Kingma, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.
- [43] J. Gu and C. Dong, “Interpreting super-resolution networks with local attribution maps,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 9199–9208.
- [44] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, Y. Qiao, and C. Change Loy, “EsrGAN: enhanced super-resolution generative adversarial networks,” in *Proceedings of the European conference on computer vision (ECCV) workshops*, 2018, pp. 0–0.
- [45] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, “The unreasonable effectiveness of deep features as a perceptual metric,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 586–595.